4.5평 김밥하우스 POS 시스템

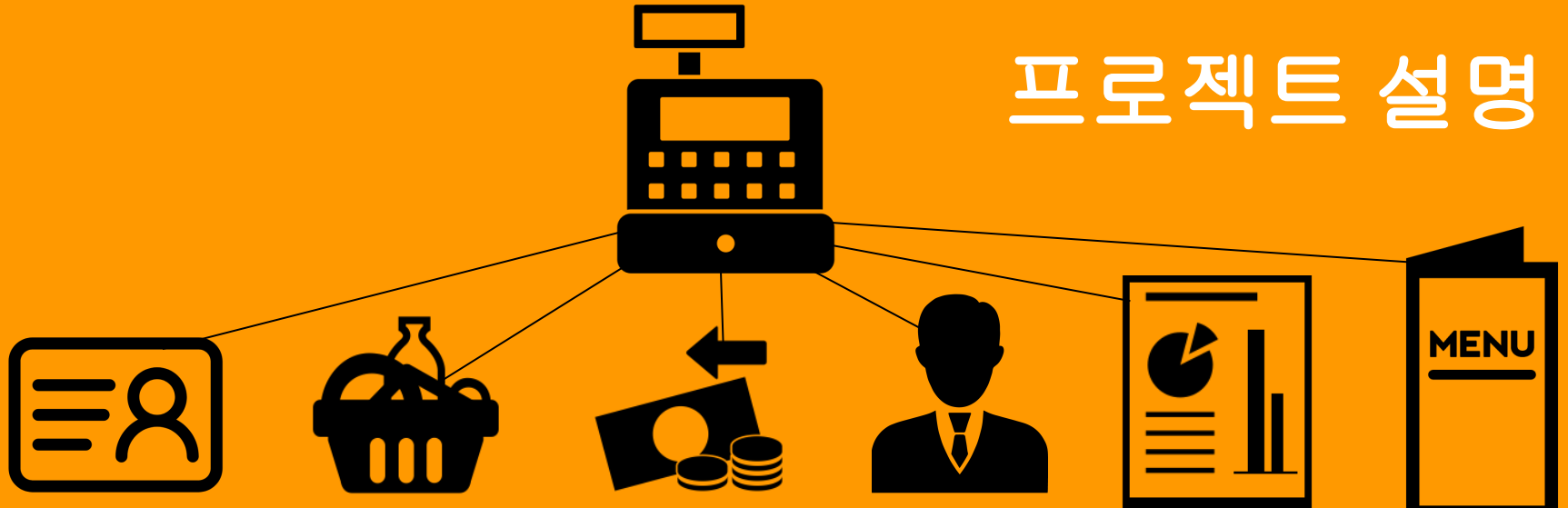
테이블 5개짜리 김밥맛집!!
다양한 김밥을 팔고 있습니다~

<우당당탕 조>
김한솔 | 강다은 | 백민종 | 주예리나 | 김다애 | 김희수

목차

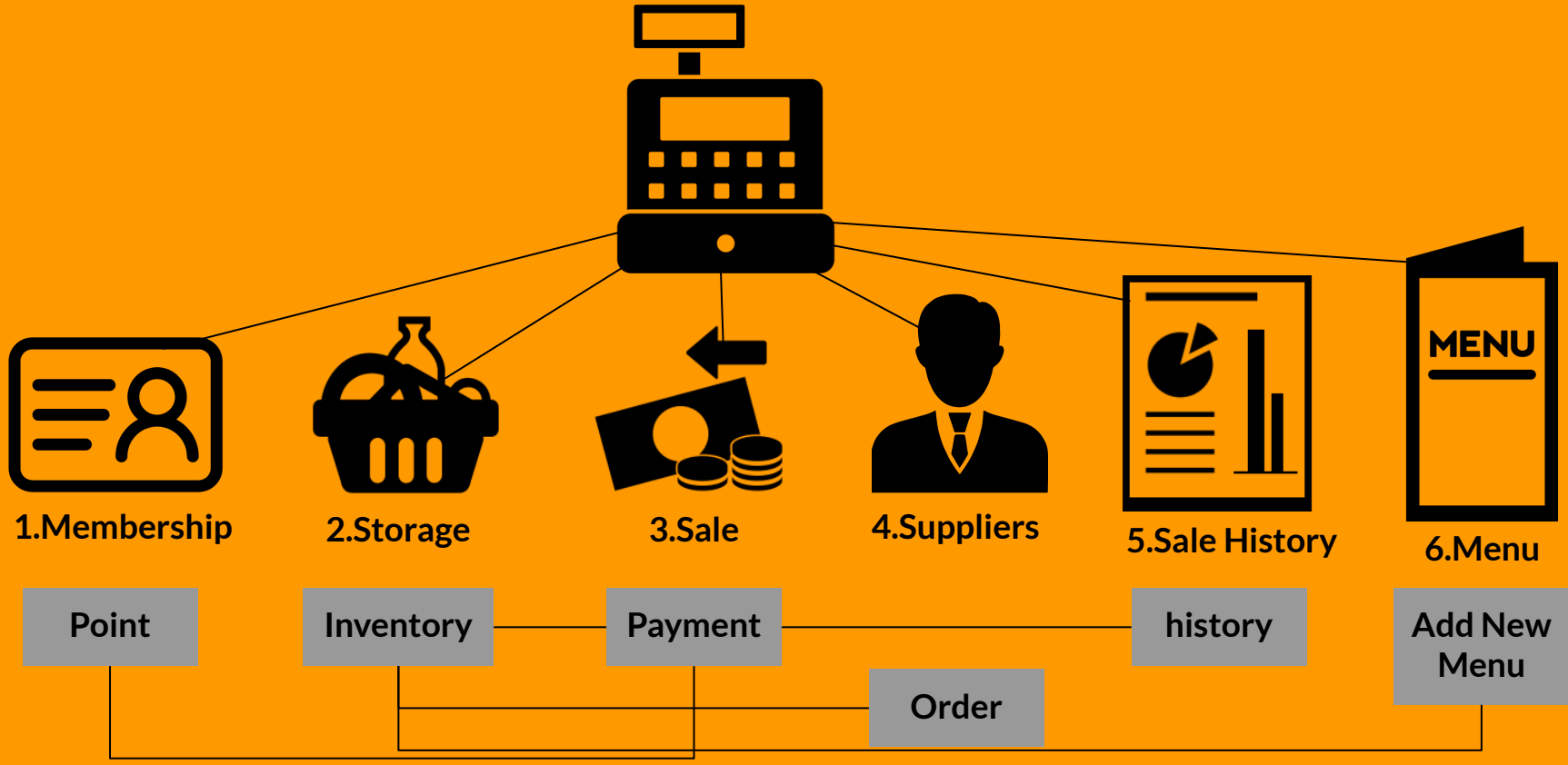
1. 전체흐름
2. 개별 기능 소개
    - 기능
    - 발생한 문제 & 해결방법
    - 개선방향
3. 전체시연

프로젝트 설명



포인트 재고 결제 history 신메뉴

발주


MENU
1.Membership
2.Storage
3.Sale
4.Suppliers
5.Sale History
6.Menu
Point
Inventory
Payment
history
Add New Menu
Order

1.회원관리

김한솔

구현기능

- 회원 추가
- 회원 정보 수정
- 회원 삭제
- 회원 정보 출력
- 결제 시 포인트 적립
- 결제 시 포인트 사용
- 포인트 정책 수립

구현방법


포인트 관리 기능
회원 관리 기능
Singleton
SellManager
포인트 사용 시 연동
PointManager
- 포인트 적립 메서드
- 포인트 사용 메서드
- 포인트 누적 점수에 따른 회원 등급 변경 메서드
Member
- 이름
- 전화번호
- 회원번호
- 레벨
- 포인트
- 회원정보 메서드
Singleton
MemberManager
- 회원정보 저장 (hashset 사용)
- 회원정보 출력
- 회원정보 삭제

기능 시연: 멤버관리

```java
// 1.입력
public void addMember() {
    System.out.print("이름: ");
    String name = util.Util.scan.nextLine();
    System.out.print("전화번호: ");
    String phoneNum = util.Util.scan.nextLine();

    Member member = new Member(name, phoneNum);

    hashSet.add(member);
    System.out.println("회원정보가 등록되었습니다.");
}
```

```java
// 2. 수정
public void changeMember() {

    System.out.print("검색할 회원의 전화번호: ");
    String phoneNum = util.Util.scan.nextLine();

    Iterator<Member> ir = hashSet.iterator();

    while (ir.hasNext()) {
        Member temp = ir.next();
        if (temp.getPhoneNum().equals(phoneNum)) {
            System.out.print("어떤 사항을 수정하시겠습니까? 1. 이름 |2. 전화번호");
            int choice = util.Util.scan.nextInt();
            util.Util.scan.nextLine();
            switch (choice) {
            case 1:
                System.out.print("이름: ");
                String rename = util.Util.scan.nextLine();
                temp.setName(rename);
                return;

            case 2:
                System.out.print("전화번호: ");
                String rePhoneNum = util.Util.scan.nextLine();
                temp.setPhoneNum(rePhoneNum);
                return;
            }
        }

    }
    System.out.println("회원정보가 존재하지 않습니다.");
}
```

기능 시연: 멤버관리

```java
// 3. 삭제
public void removeMember() {

    System.out.print("검색할 회원의 전화번호: ");
    String phoneNum = util.Util.scan.nextLine();

    Iterator<Member> ir = hashSet.iterator();
    while (ir.hasNext()) {
        Member temp = ir.next();
        if (temp.getPhoneNum().equals(phoneNum)) {
            hashSet.remove(temp);
            System.out.println("정보가 삭제되었습니다.");
            return;
        }
    }
    System.out.println("회원정보가 존재하지 않습니다.");
    return;
}
```

```java
// 4. 데이터 보여주기
public void showData() {
    System.out.print("검색할 회원의 전화번호: ");
    String phoneNum = util.Util.scan.nextLine();

    Iterator<Member> ir = hashSet.iterator();
    while (ir.hasNext()) {
        Member temp = ir.next();
        if (temp.getPhoneNum().equals(phoneNum)) {
            temp.showData();
            return;
        }
    }
    System.out.println("회원정보가 존재하지 않습니다.");
}
```

기능 시연: 멤버관리

```java
// 5. 포인트 보여주기(결제 시 사용)
public String showPoint() {
    System.out.print("검색할 회원의 전화번호: ");
    String phoneNum = util.Util.scan.nextLine();

    Iterator<Member> ir = hashSet.iterator();
    while (ir.hasNext()) {
        Member temp = ir.next();
        if (temp.getPhoneNum().equals(phoneNum)) {
            return "현재 포인트는 " + temp.getPointNum() + "입니다.";
        }
    }
    return "회원정보가 존재하지 않습니다.";
}
```

```java
// 6. 모든 데이터 보여주기
public void showAll() {
    for (Member member : hashSet) {
        member.showData();
    }
    System.out.println();
}
```

기능 시연: 포인트 적립

```java
// 포인트 적립 계산
public int calcPoint(int price, Member member) {

    String level = member.getLevel();

    int currentPoint = member.getPointNum();

    if (level == "일반회원") {
        int addPoint = (int) (price * 0.1); // 일반회원 - 10% 적립
        currentPoint += addPoint;

    } else if (level == "VIP회원") {
        int addPoint = (int) (price * 0.2); // VIP회원 - 20% 적립
        currentPoint += addPoint;
    }
    return currentPoint;
}
```

```java
// 회원정보 체크
public void checkLevel(Member member) {

    int currentPoint = member.getPointNum();

    if (currentPoint > 1000) {
        System.out.println("VIP로 업그레이드 되었습니다.");
        member.setLevel("VIP회원");
    }
}
```

```java
// ★ 포인트 증가
public void addPoint(int price) {

    System.out.print("회원의 전화번호: ");
    String phoneNum = util.scan.nextLine();

    Iterator<Member> ir = hashSet.iterator();

    while (ir.hasNext()) {
        Member temp = ir.next();
        if (temp.getPhoneNum().equals(phoneNum)) {
            int pointNum = calcPoint(price, temp);
            temp.setPointNum(pointNum);
            System.out.println("포인트가 적립 완료되었습니다.");

            checkLevel(temp); //포인트 적립 후, 포인트 반영

            return;
        }
    }
    System.out.println("회원정보가 존재하지 않습니다.");
}
```

기능 시연: 포인트 사용

```java
//포인트 차감 계산
public int deductPoint(int price, Member member) {
    int currentPoint = member.getPointNum();

    if (currentPoint > 0) {
        System.out.print("얼마의 포인트를 사용하시겠습니까?");
        int deductPoint = util.scan.nextInt(); //사용할 포인트
        util.scan.nextLine();

        if (currentPoint >= deductPoint) { //현재 포인트 > 사용할 포인트
            price -= deductPoint; //가격 - 차감포인트
            currentPoint -=deductPoint; //현재 내 포인트 = 포인트 - 차감포인트
            member.setPointNum(currentPoint);
            System.out.println("포인트가 사용되었습니다.");

        } else {
            System.out.println("현재 포인트를 초과하였습니다.");
        }
    } else {
        System.out.println("포인트가 없습니다. 포인트를 적립하시겠습니까?");
        addPoint(price);
    }
    return price;
}
```

```java
// ★ 포인트 차감
public int usePoint(int price) {

    System.out.print("회원의 전화번호: ");
    String phoneNum = util.scan.nextLine();

    Iterator<Member> ir = hashSet.iterator();

    while(ir.hasNext()) {
        Member temp = ir.next();
        if(temp.getPhoneNum().equals(phoneNum)) {
            price = deductPoint(price, temp);
            return price;
        }
    }
    System.out.println("회원정보가 존재하지 않습니다.");
    return price;
}
```

기능 시연: 계산 시, SellManager와 연동

SellManager - void pay 메서드

개선 방향

1. 기능을 분리함으로서 코드 간결화.
2. 접근제어자의 사용
3. 예외처리

발생한 문제, 그리고 해결방법

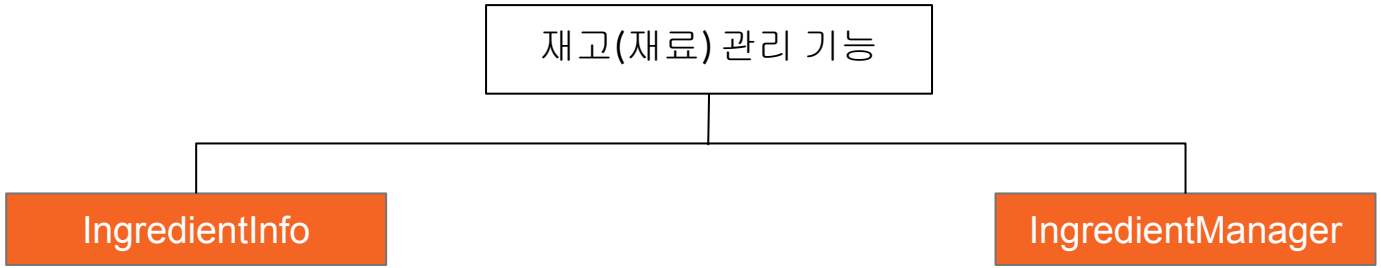
1. 회원레벨 조정 시 메서드 구동 시점에 대한 문제

2.재고관리

강다은

```
재료 이름: 김
재료 현재수량: 9
재료 최소수량: 10

재료 이름: 쌀
재료 현재수량: 100
재료 최소수량: 10

재료 이름: 단무지
재료 현재수량: 100
재료 최소수량: 10
```

구현기능



정보 저장
- 재료 이름
- 재료 현재 수량
- 재료 최소 필요 수량

기본 기능
- 재료 현황 출력
- 재고부족 알람 기능

메뉴관리와 연결:
신메뉴 추가할때 재료 추가하도록

재료정보 배열로 저장
- ArrayList

재고관리 관련 기능
- 재고관리 메뉴 출력
- 재료 검색
- 재료 추가
- 재료정보 수정
- 재료 삭제
- 재료 전체 정보 출력

기능 시연 1) 재료 검색

IngredientManager.java

```java
// 재료검색
void showInfo() {
    System.out.println("검색할 재료 이름을 입력해주세요:");
    String name = Util.sc.nextLine();

    int index = searchIndex(name);

    if (index < 0) {
        System.out.println("찾으시는 재료가 없습니다, 다시 검색해주세요.");
    } else {
        ingredientList.get(index).showInfo();
    }
}
```

```
1.재료검색|2.재료수정|3.재료삭제|4.재료전체출력|5.이전메뉴
원하시는 메뉴를 선택해주세요: 1

검색할 재료 이름을 입력해주세요:
김
재료 이름: 김
재료 현재수량: 9
재료 최소수량: 10
```

이름 입력 -> 해당 index리턴

기능 시연 2) 수정, 삭제

IngredientManager.java

```java
// 재료정보수정
void edit() {
    System.out.print("수정할 재료 이름을 입력해주세요: ");
    String name = Util.sc.nextLine();

    int index = searchIndex(name);

    if (index < 0) {
        System.out.println("찾으시는 재료가 없습니다, 다시 검색해주세요.");
    } else {
        System.out.print("[수정]이름: ");
        name = Util.sc.nextLine();
        System.out.print("[수정]현재수량: ");
        int num = Util.sc.nextInt();
        System.out.print("[수정]최소수량: ");
        int minNum = Util.sc.nextInt();

        ingredientList.remove(index);
        ingredientList.add(index, new IngredientInfo(name, minNum, num));
        System.out.println("수정되었습니다!");
    }
}
```

```
1.재료검색|2.재료수정|3.재료삭제|4.재료전체출력|5.이전메뉴
원하시는 메뉴를 선택해주세요: 2

수정할 재료 이름을 입력해주세요: 김
[수정]이름: 김2
[수정]현재수량: 20
[수정]최소수량: 100
수정되었습니다!

1.재료검색|2.재료수정|3.재료삭제|4.재료전체출력|5.이전메뉴
원하시는 메뉴를 선택해주세요: 4

재료 이름: 김2
재료 현재수량: 20
재료 최소수량: 100

재료 이름: 쌀
재료 현재수량: 100
재료 최소수량: 10

재료 이름: 단무지
재료 현재수량: 100
재료 최소수량: 10
```

기능 시연 2) 수정, 삭제

IngredientManager.java

```
// 재료삭제
void delete() {
    System.out.print("검색할 재료 이름을 입력해주세요: ");
    String name = Util.sc.nextLine();

    int index = searchIndex(name);

    if (index < 0) {
        System.out.println("찾으시는 재료가 없습니다, 다시 검색해주세요.");
    } else {
        ingredientList.remove(index);
        System.out.println("삭제되었습니다!");
    }
}
```

```
1.재료검색|2.재료수정|3.재료삭제|4.재료전체출력|5.이전메뉴
원하시는 메뉴를 선택해주세요: 3

검색할 재료 이름을 입력해주세요: 쌀
삭제되었습니다!

1.재료검색|2.재료수정|3.재료삭제|4.재료전체출력|5.이전메뉴
원하시는 메뉴를 선택해주세요: 4

재료 이름: 김2
재료 현재수량: 20
재료 최소수량: 100

재료 이름: 단무지
재료 현재수량: 100
재료 최소수량: 10
```

기능 시연 3) 출력

IngredientInfo.java

```
//개별 재료 현황 출력
void showInfo() {
    System.out.printf("재료 이름: %s\n재료 현재수량: %s\n",name,num);
    System.out.println("재료 최소수량: "+minNum);
    System.out.println();
}
```

IngredientManager.java

```
// 재료 전체 정보 출력
void showAll() {
    for (int i = 0; i < ingredientList.size(); i++) {
        ingredientList.get(i).showInfo();
    }
}
```

기능 시연 4) 재고부족알림

메인 메소드

```
public static void main(String[] args) {

    while (true) {
        for(int i=0; i<im.ingredientList.size(); i++) {
            im.ingredientList.get(i).alarm();
        }

        System.out.println("POS");
        System.out.println("1.재고 |2.판매 |3.거래처 |4.메뉴");

        int select = sc.nextInt();

        switch (select) {
        case 1:
            im.showMenu();
            break;
        case 2:
            sm.selectSellMenu();
            break;

        case 3:
            supm.printMenu();
            break;

        case 4:
            mm.MenuManage();
            break;

        default:
            System.out.println("올바르게 입력해\n");
            break;
```

포스 사용 시작 전,
재고먼저 조사하여
부족한 재료 알려준다

```
------------------------
*[김]재고부족*, 발주가 필요합니다.
현재 수량: 9
1개 부족!
------------------------
거래처 이름: 김 가게
거래처 번호: 0299565214
거래처 주소: 서울시 종로구 무슨동 어디
거래처 품목: 김
------------------------
```

기능 시연 5) 메뉴 등록 -> 메뉴에 쓰이는 재료도 함께 추가

IngredientManager.java

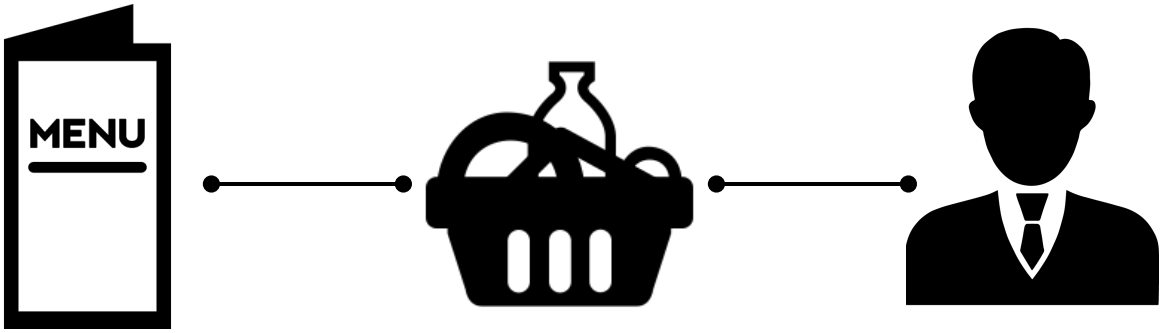
```
// 재료추가
static IngredientInfo add() {

    IngredientInfo ingredientInfo = new IngredientInfo();

    System.out.print("[추가]재료 이름: ");
    String name = Util.sc.nextLine();

    int index = searchIndex(name);

    if (!(index < 0)) {
        System.out.println("중복된 이름이 있습니다, 다시 입력해주세요");
    } else {
        System.out.print("[추가]현재수량: ");
        int num = Util.sc.nextInt();
        System.out.print("[추가]최소수량: ");
        int minNum = Util.sc.nextInt();

        ingredientList.add(new IngredientInfo(name, minNum, num));
        int i = searchIndex(name);
        System.out.println(i);
        ingredientInfo = ingredientList.get(i);
        System.out.println("추가되었습니다!");
    }

    return ingredientInfo;
}
```

MenuManager.java

```
Util.sc.nextLine();
IngredientInfo ingredientInfo = IngredientManager.add();

MenuList.add(new Menu(name, price, ingredientInfo));
```

재료관리 매니저의 재료객체
-> 메뉴관리매니저의 add()
-> MenuList배열에 넣기

발생한 문제, 그리고 해결방법



Menu

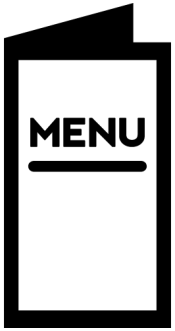
메뉴 주문 -> 해당 메뉴에 쓰이는 재료가
자동으로 감소 해야한다!

Ingredient

Supplier

현재 재료 < 최소 필요 재료
“재고부족” 알람과 동시에
해당품목 취급 거래처 정보를
불러와야 한다!

발생한 문제, 그리고 해결방법



MenuIngredient.java

Menu.java

```
public void ingredientMinus() {
    int index1 = IngredientManager.searchIndex("김");
    IngredientManager.ingredientList.get(index1).setNum(IngredientManager.ingredientList.get(index1).getNum() - 1);
    int index2 = IngredientManager.searchIndex("쌀");
    IngredientManager.ingredientList.get(index2).setNum(IngredientManager.ingredientList.get(index2).getNum() - 2);
    int index3 = IngredientManager.searchIndex("단무지");
    IngredientManager.ingredientList.get(index3).setNum(IngredientManager.ingredientList.get(index3).getNum() - 1);
    cnt++;
}
```

모든 메뉴에 쓰이는 기본 재료 -

```
public void ingredientMinus() {
    super.ingredientMinus();
    info.setNum(info.getNum() - 1);
}
```

각 메뉴에 쓰이는 개별 재료 -

발생한 문제, 그리고 해결방법

IngredientInfo.java

```
//재고가 최소수량이하로 떨어졌을시 -> '재고부족'알람
void alarm() {
    if(this.num<this.minNum) {
        System.out.println("------------------------");
        System.out.println("*["+name+"]재고부족*, 발주가 필요합니다.");
        System.out.print("현재 수량: "+num);
        System.out.println(" ("+(minNum-num)+"개 부족)");
        System.out.println("\n해당 품목 거래처 정보)");

        ////거래처관리 for문 -> equals(name) -> 거래처 showData
        for(int i=0; i<SupplierManager.si.size(); i++) {
            if(SupplierManager.si.get(i).getAddIngre().equals(name)) {
                SupplierManager.si.get(i).showData();
                System.out.println("------------------------");
            }
        }
    }
}
```

```
------------------------
*[김]재고부족*, 발주가 필요합니다.
현재 수량: 9
1개 부족!
------------------------
거래처 이름: 김 가게
거래처 번호: 0299565214
거래처 주소: 서울시 종로구 무슨동 어디
거래처 품목: 김
------------------------
```

개선 방향

1. **재료저장 세분화**
   - 기본재료가 더 추가되어야 한다면?
   - 추가재료가 2개 이상이면?
2. **재료 그룹핑**
   - 재료 품목 카테고리 (ex. 야채류, 고기류, 면류)
3. **재고 <-> 거래처 연결**
   - 부족 알람기능 -> 발주로 연결: 단순히 거래처 정보를 출력해주는 것이 아니라 발주로 연결
   - 재료 수정 -> 거래처에 저장된 재료정보도 함께 수정

3. 판매관리

백민종

구현기능

```
Guest[] TABLE = new Guest[5];
```

판매 기능

SellManager

주문 기능
- MenuList에 저장된 내용 출력
- 선택한 메뉴를 orderList에 저장
- 메뉴의 수량 만큼 재료 차감
- orderList를 받아 Guest 생성자 호출
- TABLE에 Guest 객체 저장

계산 기능
- 결제 후 해당 TABLE[] = null
- 받을 금액만큼 잔고 증가

Guest

테이블에 앉을 손님 객체
- 메뉴 이름
- 메뉴 수량
- 가격

SellManager.java

기능 시연

```
// 시작 메뉴
void selectSellMenu() {
    while (true) {

        System.out.println("판매관리");
        System.out.println("1. 주문 ¦ 2. 결제 ¦ 3.테이블 정보 ¦ 4. 이전 메뉴");

        select = sc.nextInt();

        switch (select) {
        case 1:

            // 주문을 하기 전 자리 확인.
            if (selectTable() == false) {
                System.out.println("자리가 없어.");
            } else {
                order();
            }
            break;
        case 2:
            pay();
            break;
        case 3:
            showTable();
            break;
        case 4:
            return;
        }
    }
}
```

```
// 빈 자리를 찾는 메서드
boolean selectTable() {
    boolean c = false;
    for (int i = 0; i < TABLE.length; i++) {
        if (TABLE[i] == null) {
            c = true;
        }
    }
    return c;
}
```

SellManager.java

기능 시연

```java
// 주문 받는 메서드.
void order() {
    int i;

    // 추가 주문을 하면 계속해서 반복함.
    do {
        System.out.println("-----메뉴-----");
        for (i = 0; i < MenuManager.MenuList.size(); i++) {
            System.out.print(i + 1 + ". " + MenuManager.MenuList.get(i).getName() + " ");
        }
        System.out.println();

        select = sc.nextInt();

        m = MenuManager.MenuList.get(select - 1);

        check = whatOrderNum(m);

        orderList.add(m);

    } while (check == false);

    addTable();
}
```

다른 메뉴도 같이 주문할 때 false 반환
주문을 그만할 때 true 반환

반환 값이 true일 때 반복문을 빠져나온다.

```java
// 주문 메서드
// 다른 메뉴도 주문할 때 false 값을 반환.
boolean whatOrderNum(Menu m) {
    boolean addOrder = false;
    System.out.println("몇 개 주문해 ?");
    select = sc.nextInt();
    if (select < 1) {
        System.out.println("1개 이상 주문해");

    }

    else {

        for (int i = 0; i < select; i++) {
            m.ingredientMinus();

        }
    }
    System.out.println("다른 메뉴도 주문해?");
    System.out.println("1. 응, 2. 아니");
    select = sc.nextInt();

    switch (select) {

    case 1:
        addOrder = false;
        break;

    case 2:
        System.out.println("주문 완료.");
        addOrder = true;
        break;
    }
    return addOrder;
}
```

SellManager.java

기능 시연

```
// 결제 메서드
void pay() {
    System.out.println("몇 번 테이블 계산 할거야?");
    select = scan.nextInt();

    int tableNum = select - 1;
    if (select < TABLE.length + 1 && select > 0) {
        if (!(TABLE[tableNum] == null)) {
            int totalPrice = TABLE[tableNum].getTotalPrice();
            System.out.println(select + "번 테이블");
            System.out.println("결제 금액 : " + TABLE[tableNum].getTotalPrice());
            System.out.println("회원이야?");
            System.out.println("1. 응, 2. 아니");
            select = scan.nextInt();

            scan.nextLine();
```

```
// 회원정보를 검색 후 포인트 적립 / 사용
switch (select) {
case 1:
    String message = MemberManager.getManager().showPoint();
    System.out.println(message);
    if (message == "회원정보가 존재하지 않습니다.") {
        System.out.println("가입하시겠습니까? 1.예 2.아니오 ");
        select = scan.nextInt();
        scan.nextLine();

        switch (select) { // 회원가입 여부에 대한 대답
        case 1:
            MemberManager.getManager().addMember();
            PointManager.getManager().addPoint(totalPrice);
            break;

        case 2:
            break;
        }

    } else {
        System.out.println("1. 포인트 사용 ¦ 2. 포인트 적립");
        select = scan.nextInt();
        scan.nextLine();
        switch (select) {
        case 1:
            totalPrice = PointManager.getManager().usePoint(totalPrice);
            TABLE[tableNum].setTotalPrice(totalPrice);
            MyPOS.money += TABLE[tableNum].getTotalPrice();
            break;

        case 2:
            PointManager.getManager().addPoint(totalPrice);
            break;
        }
    }
    break;
```

SellManager.java

기능 시연

```
        case 2:
            System.out.println("회원가입 하시겠습니까?");
            System.out.println("1. 응 ¦ 2. 아니");
            select = scan.nextInt();
            scan.nextLine();

            if (select == 1) {
                MemberManager.getManager().addMember();
                PointManager.getManager().addPoint(totalPrice);
                MyPOS.money += TABLE[tableNum].getTotalPrice();

            } else if (select == 2) {
                MyPOS.money += TABLE[tableNum].getTotalPrice();
                break;

            }
        }

        for (int i = 0; i < TABLE[tableNum].order.size(); i++) {

            // 히스토리 객체 생성

            long payAmount = (TABLE[tableNum].foodCnt[i] * TABLE[tableNum].order.get(i).getPrice());

            HistoryManager.insertHistory(TABLE[tableNum].payNum, LocalDateTime.now(),
                    TABLE[tableNum].order.get(i).getName(), TABLE[tableNum].foodCnt[i], payAmount, "123");
            ;


        System.out.println("결제 완료");
        System.out.println("잔고 : " + MyPOS.money);
        TABLE[tableNum] = null;

    } else {
        System.out.println("없는 테이블");

    }
```

결제 기록을 객체화 해서 저장하는 메서드

Guest 주문내역에 있는 객체의 수만큼 반복

개선 방향

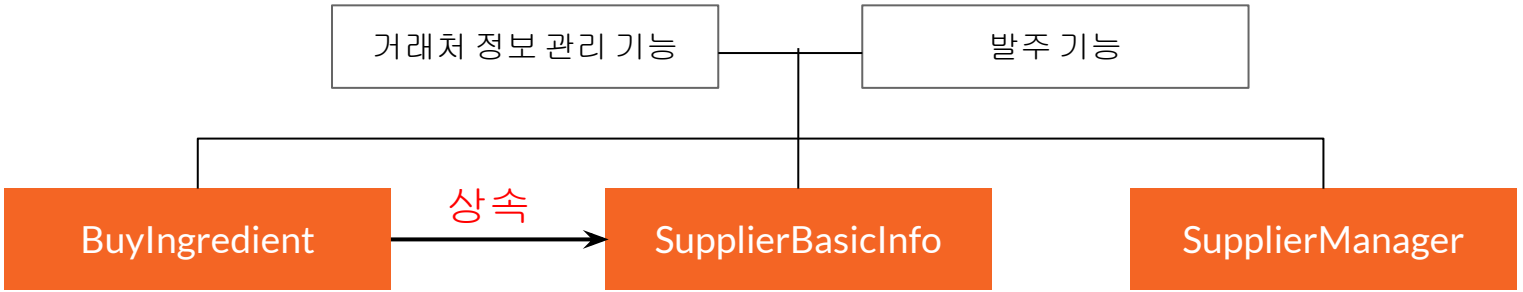
1. **pay() 메서드 간소화**
   - case 마다 다른 메서드 호출

2. **재료 수가 - 됐을 때 주문 취소 기능**
   - 입력한 값 만큼 ingredientMinus()을 수행하다가 취소.
   - 호출 전 상태로 재료 수 초기화

4.거래처관리

주예리나

구현 기능



기본 정보
- 구매할 재료이름
- 구매할 재료 가격
- 구매할 재료 원가

기능
- 발주 재료 정보 출력

기본 정보
- 거래처 이름
- 거래처 번호
- 거래처 주소

기능
- 거래처 정보 출력

거래처 정보 저장
<ArrayList>저장

거래처 정보 관리
- 정보 등록 / 검색
- 정보 수정/삭제
- 정보 출력
-

발주 기능
- 현재 발주 품목 출력
- 재료 주문 - 현재고 추가

SupplierManager
거래처 등록 기능

기능 시연

```
// 거래처 등록 기능
void insertSupplier() {

    String name;
    String phoneNum;
    String address;
    String ingName;
    int ingPrice;

    System.out.println("거래처 이름을 입력하세요.");
    name = Util.sc.nextLine();
    System.out.println("거래처 전화번호를 입력하세요.");
    phoneNum = Util.sc.nextLine();

    System.out.println("거래처 주소를 입력하세요.");
    address = Util.sc.nextLine();
    System.out.println("거래할 품목을 적어주세요.");
    ingName = Util.sc.nextLine();
    System.out.println("거래 품목의 원가를 지정해주세요.");
    ingPrice = Util.sc.nextInt();
    Util.sc.nextLine();

    si.add(new SupplierBasicInfo(name,phoneNum,address,ingName,ingPrice));
}
```

```
이름입력
거래처 이름을 입력하세요.
이름
거래처 전화번호를 입력하세요.
01023456789
거래처 주소를 입력하세요.
인천
거래할 품목을 적어주세요.
쌀
거래 품목의 원가를 지정해주세요.
1000
================================
현재 거래처 현황입니다.

================================
거래처 이름: 이천 쌀 전문
거래처 번호: 01099342242
거래처 주소: 서울시 종로구 어디어디
거래처 품목: 쌀
================================
================================
거래처 이름: 단무지 공장
거래처 번호: 01023456789
거래처 주소: 인천시 남동구 어디
거래처 품목: 단무지
================================
```

SupplierManager
거래처 삭제 기능

기능 시연

```
// 거래처 삭제
void deleteSupplier() {

    System.out.println("삭제할 거래처 번호를 입력하세요.");
    String num = Util.sc.nextLine();

    int idx = searchIndex(num);

        if(si.get(idx).phoneNum.equals(num)) {
            System.out.println("================================");
            si.get(idx).showData();
            System.out.println("================================");

            System.out.println("삭제하시겠습니까? \n1. 네 / 2. 아니오.");
            int select = Util.sc.nextInt();

            switch(select) {
                case 1:
                    si.remove(si.get(idx));
                    System.out.println("삭제를 완료했습니다.\n");
                    break;
                case 2:
                    System.out.println("취소했습니다.");
                    return;
                default :
                    System.out.println("올바른 값을 입력하세요.");
            }

        } else {

            System.out.println("다시 검색해주세요.\n");
            return;

        }
    }
```

```
c
삭제할 거래처 번호를 입력하세요.
01012345678
================================
거래처 이름: 양반김 상회
거래처 번호: 01012345678
거래처 주소: 서울시 종로구 광장시장
거래처 품목: 김
================================
삭제하시겠습니까?
1. 네 / 2. 아니오.
1
삭제를 완료했습니다.

================================
현재 거래처 현황입니다.

================================
거래처 이름: 이천 쌀 전문
거래처 번호: 01099342242
거래처 주소: 서울시 종로구 어디어디
거래처 품목: 쌀
================================
================================
거래처 이름: 단무지 공장
거래처 번호: 01023456789
거래처 주소: 인천시 남동구 어디
거래처 품목: 단무지
================================
```

SupplierManager
거래처 수정 기능

기능 시연

```
// 거래처 수정
void modifySupplier() {

    System.out.println("수정할 거래처 번호를 입력하세요.\n");
    String num = Util.sc.nextLine();

    int idx = searchIndex(num);

        if(idx>0) {

            System.out.println("================================");
            si.get(idx).showData();
            System.out.println("================================");

            System.out.println("수정할 목록을 선택하세요.\n1. 거래처 이름 / 2. 거래처 주소 / 3. 거래처 번호");
            int select = Util.sc.nextInt();
            Util.sc.nextLine();

            switch(select){

                case 1:
                    System.out.println("수정할 거래처 이름을 입력하세요.\n");
                    String name = Util.sc.nextLine();
                    si.get(idx).setName(name);
                    break;

                case 2:
                    System.out.println("수정할 거래처 주소를 입력하세요.\n");
                    String address = Util.sc.nextLine();
                    si.get(idx).setAddress(address);
                    break;

                case 3:
                    System.out.println("수정할 거래처 번호를 입력하세요.\n");
                    String phoneNum = Util.sc.nextLine();
                    si.get(idx).setPhoneNum(phoneNum);
                    break;

                default:
                    System.out.println("올바른 값을 넣어주세요.\n");

            }
        System.out.println("수정을 완료했습니다.\n");

        } else {

            System.out.println("해당 정보를 가진 거래처가 존재하지 않습니다.\n");
            return;

        }
    }
```

```
수정할 거래처 번호를 입력하세요.

01023456789
================================
거래처 이름: 단무지 공장
거래처 번호: 01023456789
거래처 주소: 인천시 남동구 어디
거래처 품목: 단무지
================================
수정할 목록을 선택하세요.
1. 거래처 이름 / 2. 거래처 주소 / 3. 거래처 번호
3
수정할 거래처 번호를 입력하세요.

01093016267
수정을 완료했습니다.

================================
거래처 이름: 이천 쌀 전문
거래처 번호: 01099342242
거래처 주소: 서울시 종로구 어디어디
거래처 품목: 쌀
================================
================================
거래처 이름: 단무지 공장
거래처 번호: 01093016267
거래처 주소: 인천시 남동구 어디
거래처 품목: 단무지
================================
```

SupplierManager
발주하기 기능

기능 시연

```
// 발주 기능
void orderSupplier() {
    int num = 0;
    int idx = -1;
    System.out.println("[발주 관리]");
    System.out.println("발주할 재료의 이름을 입력하세요.\n");
    String name = Util.sc.nextLine();


    for(int i=0;i<IngredientManager.ingredientList.size();i++) {

        if(IngredientManager.ingredientList.get(i).getName().equals(name)) {
            idx = i;
        }
    }


    if(idx>0) {
        System.out.println("주문 수량을 입력해주세요.");

        // 재료 수량 추가
        num = Util.sc.nextInt();
        System.out.println("-------------------------");
        System.out.println("예상 재고 수량 : "+(int)(IngredientManager.ingredientList.get(idx).getNum()+(num))+"\n주문 하시겠습니까? 1. 네 2.
아니오\n");
        System.out.println("-------------------------");
        int select = Util.sc.nextInt();
        switch(select) {
            case 1:
                IngredientManager.ingredientList.get(idx).setNum(IngredientManager.ingredientList.get(idx).getNum()+num);
                System.out.println("발주를 완료했습니다.\n");
                break;
            case 2:
                System.out.println("취소했습니다.\n");
                break;
        }
    } else {
            System.out.println("품목이 일치하지 않습니다. 다시 입력해주세요.\n");
    }
```

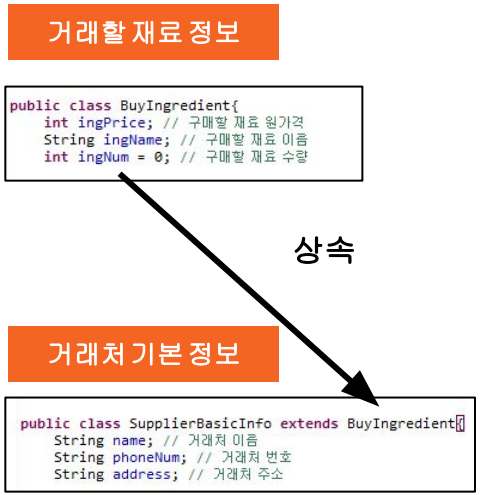
가지고 있는 재고 수량에 주문받은 수량을 더함.
추가될 재고 수량을 확인 가능.

재료 정보 객체에서 재료의 수량을 가져와서 받은 수량과 현재 수량을 더해서 현재 수량을 수정함.

```
[발주 관리]
발주할 재료의 이름을 입력하세요.

쌀
주문 수량을 입력해주세요.
10
-------------------------
예상 재고 수량 : 110
주문 하시겠습니까? 1. 네 2. 아니오

-------------------------
1
발주를 완료했습니다.

======================
재료 이름: 김
재료 현재수량:      100
재료 최소수량:      10
======================
======================
재료 이름: 쌀
재료 현재수량:      110
재료 최소수량:      10
======================
======================
재료 이름: 단무지
재료 현재수량:      100
재료 최소수량:      10
======================
```

발생한 문제 & 해결

거래처의 품목을 현 재고 관리로 넘겨줄 수 없는 상황.




```
SupplierBasicInfo(String name, String phoneNum,String address, String ingName, int ingPrice) {
    super(ingName,ingPrice);
    this.name = name;
    this.phoneNum = phoneNum;
    this.address = address;
}


// 거래 재료 이름 넘기기
String getIngName() {
    return ingName;
}
```

프로젝트 개선 방향

1. 재료와 거래처 품목 연결

재료 추가 시 거래처 품목 동기화

➔ 거래처와 함께 거래 품목과 원가를 지정
➔ 재고 키워드를 통해서 찾아야 하는 번거로움

재료 관리에서 알림과 함께 발주하는 기능 추가

2. 거래처 등록 -> 전화번호 중복 / 키워드 중복

HashSet을 통해 중복 제거 해결

3. 접근 제어자로 보안 구현

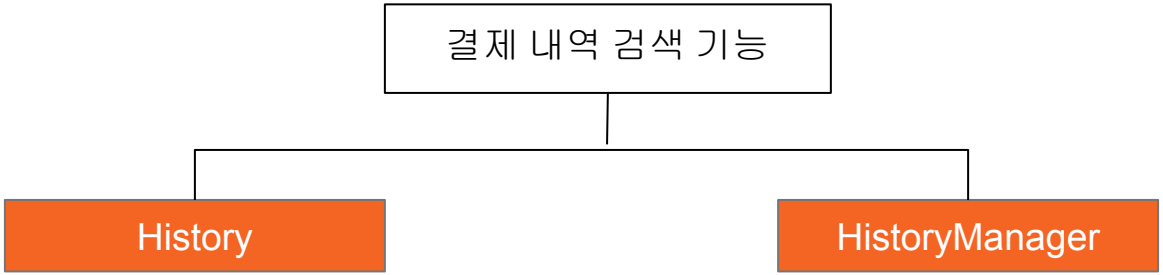
5. 판매내역

김다애

구현 기능 ٩(*´ □ `*)۶

- 1. 오늘의 결제내역
- 2. 일별 결제내역 검색
- 3. 월별 결제내역 검색
- 4. 회원별 결제내역
- 5. 모든 결제내역 보기

구현 기능 ٩(*´ □ `*)۶

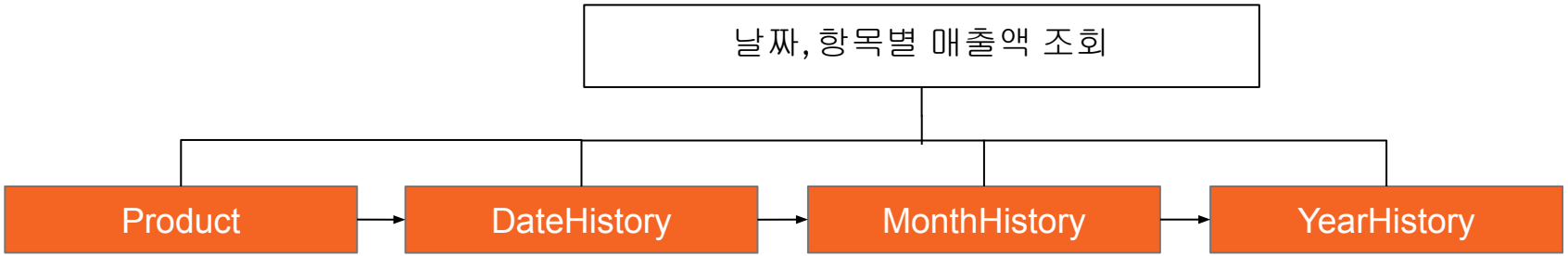


결제 정보 저장
- 결제 시각
- 결제 항목
- 결제 개수
- 결제 금액
- 결제한 사람

기본 기능
- 결제 내역 한 줄 출력

결제 정보 검색기능
- 오늘의 결제내역 검색
- 특정 일의 결제내역 검색
- 특정 월의 결제내역 검색
- 회원별 검색
- 전체 검색

구현(중인) 기능 (•̀ ᴗ •́)



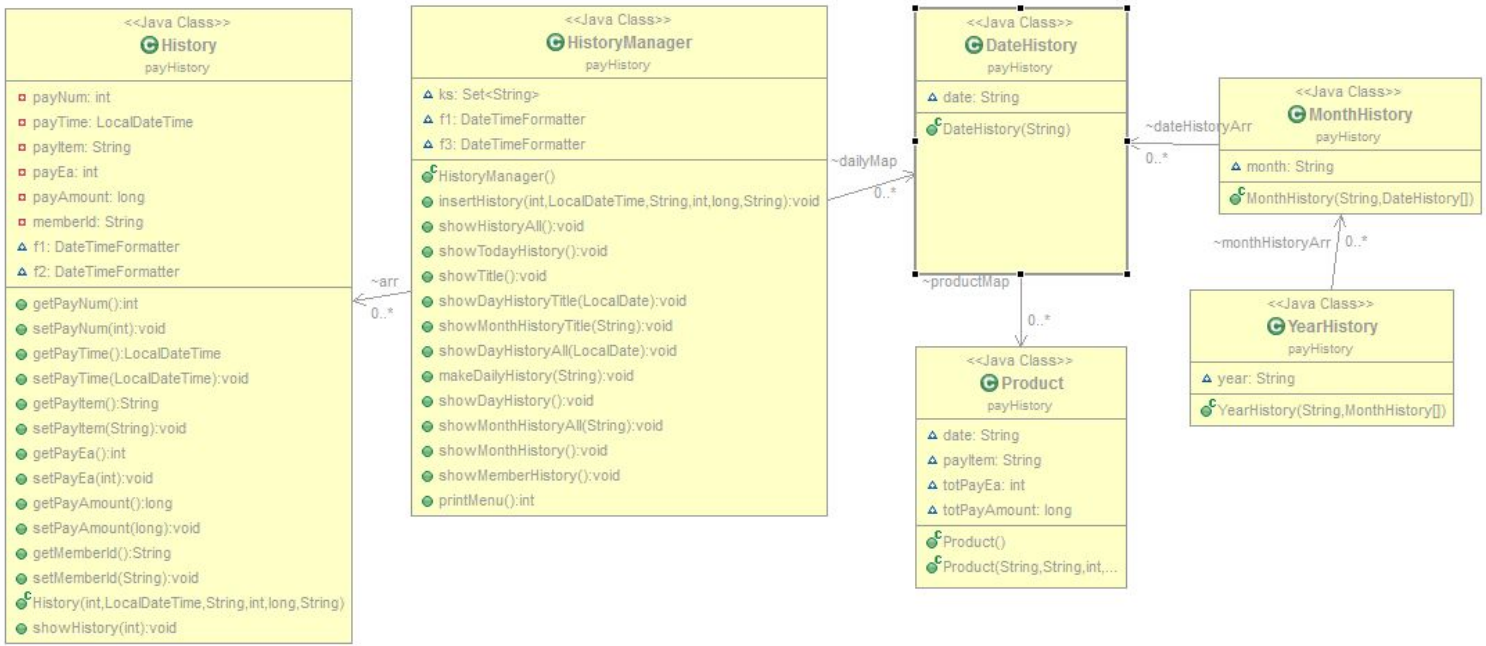
저장
- **결제 항목**
- 결제 메뉴
- 결제 개수
- 결제 시각

저장
- **결제 날짜**
- Product

저장
- **결제 월**
- DateHistory

저장
- **결제 월**
- MonthHistory

클래스로 보자면요..! (•̀ ᴗ •́)✧


<<Java Class>>
History
payHistory
payNum: int
payTime: LocalDateTime
payItem: String
payEa: int
payAmount: long
memberId: String
f1: DateTimeFormatter
f2: DateTimeFormatter
getPayNum():int
setPayNum(int):void
getPayTime():LocalDateTime
setPayTime(LocalDateTime):void
getPayItem():String
setPayItem(String):void
getPayEa():int
setPayEa(int):void
getPayAmount():long
setPayAmount(long):void
getMemberId():String
setMemberId(String):void
History(int,LocalDateTime,String,int,long,String)
showHistory(int):void
~arr
0..*
<<Java Class>>
HistoryManager
payHistory
ks: Set<String>
f1: DateTimeFormatter
f3: DateTimeFormatter
HistoryManager()
insertHistory(int,LocalDateTime,String,int,long,String):void
showHistoryAll():void
showTodayHistory():void
showTitle():void
showDayHistoryTitle(LocalDate):void
showMonthHistoryTitle(String):void
showDayHistoryAll(LocalDate):void
makeDailyHistory(String):void
showDayHistory():void
showMonthHistoryAll(String):void
showMonthHistory():void
showMemberHistory():void
printMenu():int
~dailyMap
0..*
<<Java Class>>
DateHistory
payHistory
date: String
DateHistory(String)
~dateHistoryArr
0..*
<<Java Class>>
MonthHistory
payHistory
month: String
MonthHistory(String,DateHistory[])
~monthHistoryArr
0..*
~productMap
0..*
<<Java Class>>
YearHistory
payHistory
year: String
YearHistory(String,MonthHistory[])
<<Java Class>>
Product
payHistory
date: String
payItem: String
totPayEa: int
totPayAmount: long
Product()
Product(String,String,int,...

1. 오늘의 결제내역

showTodayHistory()

```
// ================================================================
// 1. 오늘의 결제내역을 보여주는 메소드
public void showTodayHistory() {
    showDayHistoryAll(LocalDate.now());
}
// ================================================================
```

1. 오늘의 결제내역

오늘날짜
now() 메소드

showDayHistoryAll(LocalDate date)

```
// 특정 날짜의 히스토리 알아내기
// 받은걸 써야(수정해야함)
public void showDayHistoryAll(LocalDate date) {
    showDayHistoryTitle(date);
    if (arr.size() < 1) {
        System.out.println("결제 내역이 없습니다.");
    } else {
        int cnt = 0;
        int tot = 0;
        int kimbobCnt = 0;
        for (int i = 0; i < arr.size(); i++) {
            if (arr.get(i).getPayTime().getDayOfYear() == date.getDayOfYear()) {
                tot += arr.get(i).getPayAmount();
                kimbobCnt += arr.get(i).getPayEa();
                cnt++;
                arr.get(i).showHistory(cnt);
            }
        }
        System.out.println(
                "-------------------------------------------------------------------------------------------");
        System.out.println("총 매출액 :" + tot + ", " + "팔린 김밥 개수 : " + kimbobCnt);
    }

    System.out
            .println("*******************************************************************************************");
    System.out.println();
    System.out.println();
}
```

1. 오늘의 결제내역 결과화면 ヽ(ˇ³ˇ)ノ♡

```
======================================  메뉴를 선택해주세요 ======================================
1.오늘의 결제내역       2.일별 검색          3.월별 결제내역 검색    4.회원별 결제내역       5.모든 결제내역 보기
=================================================================================================
1
*******************************  [2019년 5월 29일 수요일의 결제내역]  *******************************
No      결제 시각   결제 메뉴   수량      결제금액            구매자            결제번호
-------------------------------------------------------------------------------------------------
1       오후 7:40  파김밥      5개       35000             01031311717      4       2019년 5월 29일 수요일
2       오후 8:55  야채김밥    2개       5000              01012345678      5       2019년 5월 29일 수요일
3       오후 9:17  멍멍김밥    3개       9000              01014568745      6       2019년 5월 29일 수요일
-------------------------------------------------------------------------------------------------
총 매출액 :49000, 팔린 김밥 개수 :  10
*************************************************************************************************
```

2. 특정 일의 결제내역 검색

showDayHistoryAll(LocalDate date)

원하는 날짜 입력 •ㅗ•

결제내역 출력 ʕ·ᴥ·ʔ/

```
// 특정 날짜의 히스토리 알아내기
// 받은걸 써야(수정해야함)
public void showDayHistoryAll(LocalDate date) {
    showDayHistoryTitle(date);
    if (arr.size() < 1) {
        System.out.println("결제 내역이 없습니다.");
    } else {
        int cnt = 0;
        int tot = 0;
        int kimbobCnt = 0;
        for (int i = 0; i < arr.size(); i++) {
            if (arr.get(i).getPayTime().getDayOfYear() == date.getDayOfYear()) {
                tot += arr.get(i).getPayAmount();
                kimbobCnt += arr.get(i).getPayEa();
                cnt++;
                arr.get(i).showHistory(cnt);
            }
        }
        System.out.println(
                "-----------------------------------------------------------------------------------------");
        System.out.println("총 매출액 :" + tot + ", " + "팔린 김밥 개수 : " + kimbobCnt);
    }

    System.out
            .println("*****************************************************************************************");
    System.out.println();
    System.out.println();
}
```

2. 일별 결제내역 검색 결과 ◟(˘ ³˘)◞♡

```
=====================================  메뉴를 선택해주세요 ==================================
1.오늘의 결제내역       2.일별 검색          3.월별 결제내역 검색    4.회원별 결제내역      5.모든 결제내역 보기
=============================================================================================
2
어떤 날짜의 결제 내역을 출력할까요?(숫자 8자리로 입력해주세요 ex.20190527)
20190528
*******************************  [2019년 5월 28일 화요일의 결제내역]  *******************************
No      결제 시각   결제 메뉴   수량      결제금액              구매자              결제번호
---------------------------------------------------------------------------------------------
1       오후 1:20  멸추김밥    2개       6000                01031251111       1       2019년 5월 28일 화요일
2       오후 2:50  메롱김밥    1개       7000                01011111111       2       2019년 5월 28일 화요일
3       오후 3:50  또잉김밥    2개       11000               01011111111       2       2019년 5월 28일 화요일
4       오후 4:50  메롱김밥    5개       35000               01012346554       3       2019년 5월 28일 화요일
---------------------------------------------------------------------------------------------
총 매출액 :59000, 팔린 김밥 개수 :  10
*********************************************************************************************
```

3. 특정 월의 결제내역 검색

showDayHistoryAll(LocalDate date)

원하는 월 입력 •ㅗ•

결제내역 출력 ʕ·ᴥ·ʔ/

```
// 특정 날짜의 히스토리 알아내기
// 받은걸 써야(수정해야함)
public void showDayHistoryAll(LocalDate date) {
    showDayHistoryTitle(date);
    if (arr.size() < 1) {
        System.out.println("결제 내역이 없습니다.");
    } else {
        int cnt = 0;
        int tot = 0;
        int kimbobCnt = 0;
        for (int i = 0; i < arr.size(); i++) {
            if (arr.get(i).getPayTime().getDayOfYear() == date.getDayOfYear()) {
                tot += arr.get(i).getPayAmount();
                kimbobCnt += arr.get(i).getPayEa();
                cnt++;
                arr.get(i).showHistory(cnt);
            }
        }
        System.out.println(
                "-----------------------------------------------------------------------------------------");
        System.out.println("총 매출액 :" + tot + ", " + "팔린 김밥 개수 : " + kimbobCnt);
    }

    System.out
            .println("*****************************************************************************************");
    System.out.println();
    System.out.println();
}
```

3. 월별 결제내역 검색 결과 ˛(˘ ³˘)˛♡

```
======================================== 메뉴를 선택해주세요 ========================================
1.오늘의 결제내역      2.일별 검색      3.월별 결제내역 검색   4.회원별 결제내역      5.모든 결제내역 보기
================================================================================================
3
원하는 월을 입력해주세요.(숫자 6자리로 입력해주세요 ex.201905)
201905
****************************** [201905의 상세 결제내역 ]******************************
```

No	결제 시각	결제 메뉴	수량	결제금액	구매자	결제번호	
1	오후 1:16	멸추김밥	2개	6000	01031251111	1	2019년 5월 28일 화요일
2	오후 2:46	메롱김밥	1개	7000	01011111111	2	2019년 5월 28일 화요일
3	오후 3:46	또잉김밥	2개	11000	01011111111	2	2019년 5월 28일 화요일
4	오후 4:46	메롱김밥	5개	35000	01012346554	3	2019년 5월 28일 화요일
5	오후 6:46	파김밥	5개	35000	01031311717	4	2019년 5월 29일 수요일
6	오후 8:01	야채김밥	2개	5000	01012345678	5	2019년 5월 29일 수요일
7	오후 8:23	멍멍김밥	3개	9000	01014568745	6	2019년 5월 29일 수요일
8	오후 1:16	야채김밥	9개	26000	01033337777	7	2019년 5월 30일 목요일
9	오후 2:46	그냥김밥	1개	3000	01031251111	8	2019년 5월 31일 금요일

```
총 매출액 :137000, 팔린 김밥 개수 : 30
```

4. 회원의 결제내역 검색

showMemberHistory()

원하는 회원 입력 •ㅗ•

결제내역 출력 ʕ·ᴥ·ʔノ

```java
//4. 회원의 결제내역 출력
public void showMemberHistory() {
    System.out.println("어떤 회원의 결제 내역을 출력할까요?");
    String member = Util.keyboard.nextLine();

    System.out.println(member + "님의 결제내역입니다");
    showTitle();
    int cnt = 0;
    for (int i = 0; i < arr.size(); i++) {
        if (arr.get(i).getMemberId().equals(member.trim())) {
            cnt++;
            arr.get(i).showHistory(cnt);
        }
    }
}
```

4. 회원의 결제내역 검색 결과 ◟(˘ ³˘)◞♡

```
=====================================  메뉴를 선택해주세요 ====================================
1.오늘의 결제내역       2.일별 검색          3.월별 결제내역 검색    4.회원별 결제내역      5.모든 결제내역 보기
================================================================================================
4
어떤 회원의 결제 내역을 출력할까요?
01031251753
01031251753님의 결제내역입니다
No      결제 시각    결제 메뉴    수량      결제금액            구매자              결제번호
------------------------------------------------------------------------------------------------
1       오후 2:17  그냥김밥    1개      3000            01031251753      9       2019년 6월 29일 토요일
2       오전 12:30피자김밥    1개      3000            01031251753      11      2019년 7월 3일 수요일
```

5. 전체 결제내역

showHistoryAll()

```
// 5.전체출력 메소드
public void showHistoryAll() {
    System.out.println("****************************  [저장된 결제내역 전체를 출력합니다.]  *******************************");
    System.out.println("No\t결제 시각\t결제 메뉴\t수량\t결제금액\t구매자\t\t결제번호\t결제 날짜");
    System.out
            .println("-------------------------------------------------------------------------------------------------");
    if (arr.size() < 1) {
        System.out.println("결제 내역이 없습니다.");
    } else {
        int cnt = 0;
        int tot = 0;
        int kimbobCnt = 0;

        for (int i = 0; i < arr.size(); i++) {
            cnt++;
            tot += arr.get(i).getPayAmount();
            kimbobCnt += arr.get(i).getPayEa();

            arr.get(i).showHistory(cnt);
        }
        System.out.println(
                "-------------------------------------------------------------------------------------------------");
        System.out.println("총 매출액 :" + tot + ", " + "팔린 김밥 개수 : " + kimbobCnt);

    }

}
```

5. 전체 결제내역 결과 화면 ヽ(ˇ³ˇ)ノ♡

```
=====================================  메뉴를 선택해주세요 ====================================
1.오늘의 결제내역       2.일별 검색          3.월별 결제내역 검색    4.회원별 결제내역       5.모든 결제내역 보기
=============================================================================================
5
*****************************  [저장된 결제내역 전체를 출력합니다.]  *********************************
```

No	결제 시각	결제 메뉴	수량	결제금액	구매자	결제번호	결제 날짜
1	오후 2:25	멸추김밥	2개	6000	01031251111	1	2019년 5월 28일 화요일
2	오후 3:55	메롱김밥	1개	7000	01011111111	2	2019년 5월 28일 화요일
3	오후 4:55	또잉김밥	2개	11000	01011111111	2	2019년 5월 28일 화요일
4	오후 5:55	메롱김밥	5개	35000	01012346554	3	2019년 5월 28일 화요일
5	오후 7:55	파김밥	5개	35000	01031311717	4	2019년 5월 29일 수요일
6	오후 9:10	야채김밥	2개	5000	01012345678	5	2019년 5월 29일 수요일
7	오후 9:32	멍멍김밥	3개	9000	01014568745	6	2019년 5월 29일 수요일
8	오후 2:25	야채김밥	9개	26000	01033337777	7	2019년 5월 30일 목요일
9	오후 3:55	그냥김밥	1개	3000	01031251111	8	2019년 5월 31일 금요일
10	오후 2:25	그냥김밥	1개	3000	01031251753	9	2019년 6월 29일 토요일
11	오후 3:55	먀아김밥	5개	13000	01031251111	10	2019년 7월 1일 월요일
12	오전 12:38	피자김밥	1개	3000	01031251753	11	2019년 7월 3일 수요일

```
총 매출액 :156000, 팔린 김밥 개수 :  37
```

구현 중 발생한 문제와 해결방법(•̀ᴗ•́)

- 메뉴별, 날짜별 통계

  자료구조를 배열로 하려니 복잡해짐

  해결방안 -> HashMap 사용

```
public void makeDailyHistory(String yyyyMMdd) {
    // ArrayList<Product> pro = new ArrayList<Product>();
    DateHistory dh;

    // dailyMap.containsKey(yyyyMMdd)
    if (dailyMap.get(yyyyMMdd) == null) {
        dh = new DateHistory(yyyyMMdd);

        for (int i = 0; i < arr.size(); i++) {
            String item = arr.get(i).getPayItem();// 결제내역에서 뽑아온 항목의 이름
            if(transFormat(arr.get(i).getPayTime()).equals(yyyyMMdd)) {
                Product pro = new Product(yyyyMMdd, item, arr.get(i).getPayEa(), arr.get(i).getPayAmount());
                dh.productMap.put(item, pro);

                // 생성해서 프로덕트 맵에 넣었다 아이템명, 프로덕트
                if (dh.productMap.get(item).payItem.equals(item)) {
                    dh.productMap.get(item).totPayEa += arr.get(i).getPayEa();
                    dh.productMap.get(item).totPayEa += arr.get(i).getPayAmount();
                }
                // 특정날짜, 데일리 히스토리를 데일리맵에 넣었다
                dailyMap.put(yyyyMMdd, dh);
            }
            System.out.println(dailyMap.get(yyyyMMdd).productMap.get(item).payItem);
        }
    }

}
```

구현 중 발생한 문제와 해결방법(•̀ᴗ•́)

- Date 클래스 getYear() 등 Deprecated

  -> LocalDate 사용

프로젝트의 개선방향은 무엇인가요?

```
==================================== 메뉴를 선택해주세요 ====================================
1.오늘의 결제내역      2.일별 검색      3.월별 결제내역 검색    4.회원별 결제내역      5.모든 결제내역 보기
=============================================================================================
5
****************************** [저장된 결제내역 전체를 출력합니다.] ******************************
```

No	결제 시각	결제 메뉴	수량	결제금액	구매자	결제번호	결제 날짜
1	오후 2:25	멸추김밥	2개	6000	01031251111	1	2019년 5월 28일 화요일
2	오후 3:55	메롱김밥	1개	7000	01011111111	2	2019년 5월 28일 화요일
3	오후 4:55	또잉김밥	2개	11000	01011111111	2	2019년 5월 28일 화요일
4	오후 5:55	메롱김밥	5개	35000	01012346554	3	2019년 5월 28일 화요일
5	오후 7:55	파김밥	5개	35000	01031311717	4	2019년 5월 29일 수요일
6	오후 9:10	야채김밥	2개	5000	01012345678	5	2019년 5월 29일 수요일
7	오후 9:32	멍멍김밥	3개	9000	01014568745	6	2019년 5월 29일 수요일
8	오후 2:25	야채김밥	9개	26000	01033337777	7	2019년 5월 30일 목요일
9	오후 3:55	그냥김밥	1개	3000	01031251111	8	2019년 5월 31일 금요일
10	오후 2:25	그냥김밥	1개	3000	01031251753	9	2019년 6월 29일 토요일
11	오후 3:55	먀아김밥	5개	13000	01031251111	10	2019년 7월 1일 월요일
12	오전 12:38	피자김밥	1개	3000	01031251753	11	2019년 7월 3일 수요일

```
총 매출액 :156000, 팔린 김밥 개수 : 37
```

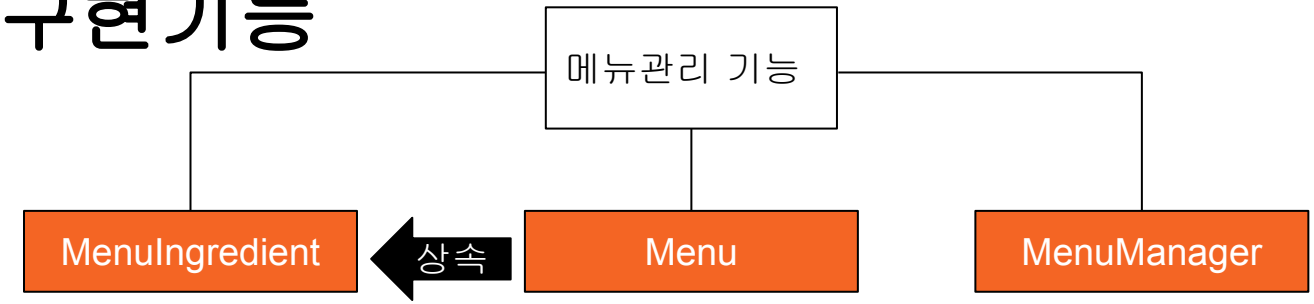
프로젝트의 개선방향은 무엇인가요?

- 통계 기능
- 매장의 인기메뉴 기능

6.메뉴관리

김희수

구현기능



- 기본 재료 정보 저장
- 메서드

  -재료 감소
  -재료 출력

- 추가 재료 정보 저장
- 메서드

  -추가재료 감소
  -추가재료 출력

- 메뉴 정보 배열저장 (Arraylist)
- 메뉴관리: 메뉴 추가,수정, 삭제,출력

— 1) 메뉴 추가

MenuManager.java

기능 시연

```
void menuAdd() {
    System.out.println("추가할 메뉴의 이름을 입력하세요 : ");
    String name = Util.sc.nextLine();

    System.out.println("추가할 메뉴의 가격을 입력하세요 : ");
    int price = Util.sc.nextInt();

    int index = searchIndex(name);

    if (!(index < 0)) {
        System.out.println("같은 이름이 존재합니다.\n 다른 이름을 입력해주세요");
    } else {

        System.out.println("새로운 메뉴이름: " + name);
        System.out.println("가격 :" + price);

        Util.sc.nextLine();
        IngredientInfo ingredientInfo = IngredientManager.add();

        MenuList.add(new Menu(name, price, ingredientInfo));
        System.out.printf("%d원인 %s메뉴가 추가되었습니다.\n", price, name);
    }
```

```
==============================
 1.메뉴 추가
 2.메뉴 수정
 3.메뉴 삭제
 4.전체 메뉴 출력
 5.종료
==============================
원하는 메뉴를 선택해주세요 :
1

추가할 메뉴의 이름을 입력하세요 :
새우김밥
추가할 메뉴의 가격을 입력하세요 :
4000
새로운 메뉴이름: 새우김밥
가격 :4000
[추가]재료 이름: 새우
[추가]현재수량: 40
[추가]최소수량: 10
2
추가되었습니다!
4000원인 새우김밥메뉴가 추가되었습니다.
```

—2) 전체 메뉴 정보 출력

기능 시연

MenuIngredient.java

메뉴의 정보를 입력 -> 기본재료(김,쌀,단무지)의 정보 함께 출력

```
void showInfo() {
    System.out.println("메뉴 이름 : " + name);
    System.out.println("메뉴 가격 : " + price);
    for (int i = 0; i < 3; i++) {
        IngredientManager.ingredientList.get(i).showInfo();
    }

}
```

Menu.java

추가재료가 필요한 메뉴 생성시
→ 추가재료 정보 출력(Override)

```
public void showInfo() {
    super.showInfo();
    int index = IngredientManager.searchIndex(info.getName());
    IngredientManager.ingredientList.get(index).showInfo();
}
```

```
=======================
 1.메뉴 추가
 2.메뉴 수정
 3.메뉴 삭제
 4.전체 메뉴 출력
 5.종료
=======================
원하는 메뉴를 선택해주세요 :
4

=======================
메뉴 이름 : 야채김밥
메뉴 가격 : 3000
=======================
재료 이름: 김
재료 현재수량: 9
재료 최소수량: 10
=======================
=======================
재료 이름: 쌀
재료 현재수량: 100
재료 최소수량: 10
=======================
=======================
재료 이름: 단무지
재료 현재수량: 100
재료 최소수량: 10
=======================
=======================
재료 이름: 야채
재료 현재수량: 2
재료 최소수량: 10
=======================
```

—2) 전체 메뉴 정보 출력

MenuManager.java

기능 시연

```
public class MenuManager {

    static ArrayList<Menu> MenuList = new ArrayList<Menu>();
    public MenuManager() {
        IngredientInfo ingre1 = new IngredientInfo("야채", 10, 2);
        IngredientManager.ingredientList.add(ingre1);
        Menu kimbab1 = new Menu("야채김밥", 3000, ingre1);
        MenuList.add(kimbab1);

  void showAllMenu() {
        System.out.println("========================");

        if (MenuList.size() > 0) {
            for (int i = 0; i < MenuList.size(); i++) {
                MenuList.get(i).showInfo();
                System.out.println("------------");
            }
        } else {
            System.out.println("등록된 메뉴정보가 없습니다.");
        }
        System.out.println("========================");
    }
```

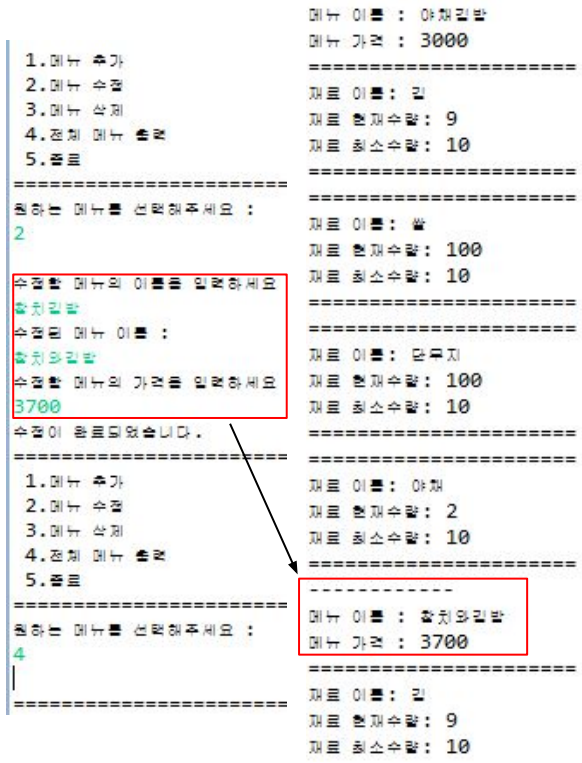
```
========================
 1.메뉴 추가
 2.메뉴 수정
 3.메뉴 삭제
 4.전체 메뉴 출력
 5.종료
========================
원하는 메뉴를 선택해주세요 :
4

========================
메뉴 이름 : 야채김밥
메뉴 가격 : 3000
========================
재료 이름: 김
재료 현재수량: 9
재료 최소수량: 10
========================
========================
재료 이름: 쌀
재료 현재수량: 100
재료 최소수량: 10
========================
========================
재료 이름: 단무지
재료 현재수량: 100
재료 최소수량: 10
========================
========================
재료 이름: 야채
재료 현재수량: 2
재료 최소수량: 10
========================
```

3) 메뉴 수정

MenuManager.java

기능 시연

```
void menuEdit() {
    System.out.println("수정할 메뉴의 이름을 입력하세요 : ");
    String name = Util.sc.nextLine();

    int index = searchIndex(name);

    if (index < 0) {
        System.out.println("메뉴의 이름이 존재하지 않습니다.");
    } else {

        System.out.println("수정된 메뉴 이름 : ");
        name = Util.sc.nextLine();
        System.out.println("수정할 메뉴의 가격을 입력하세요 : ");
        int price = Util.sc.nextInt();
        System.out.println("수정이 완료되었습니다.");

        MenuList.remove(index);
        MenuList.add(index, new Menu(name, price));
```

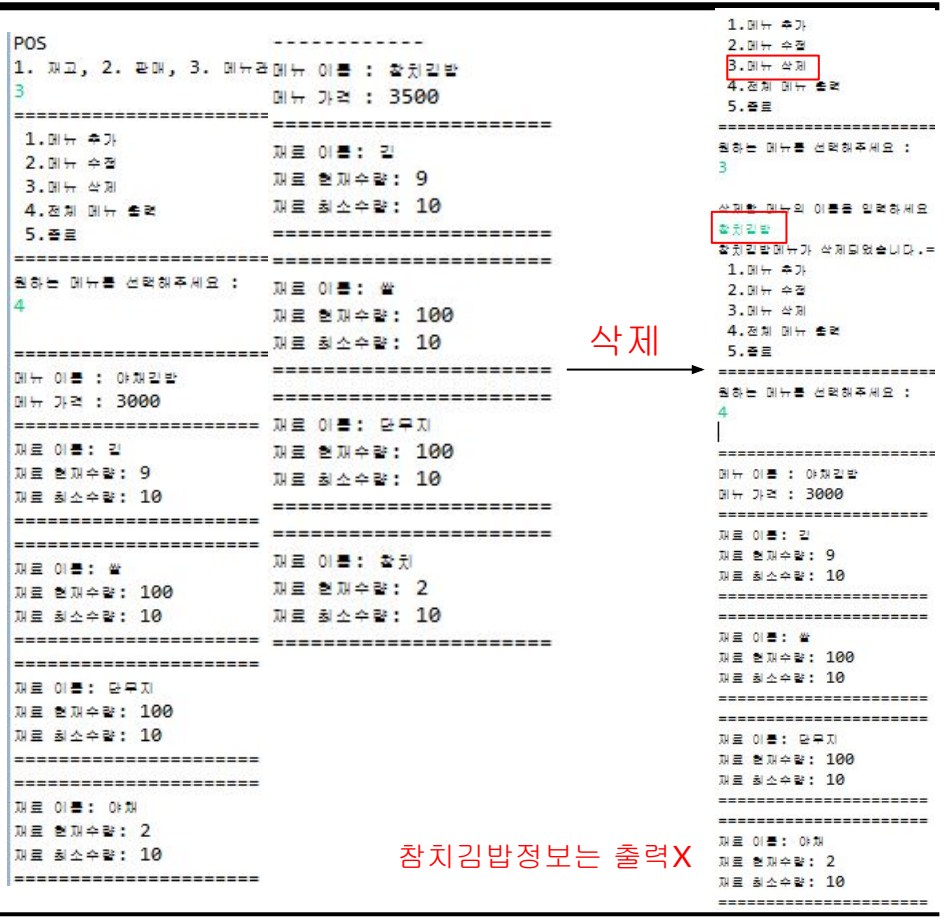


4) 메뉴 삭제

MenuManager.java

```
void menuDelete() {

    System.out.println("삭제할 메뉴의 이름을 입력하세요 : ");
    String name = Util.sc.nextLine();

    int index = searchIndex(name);

    if (index < 0) {
        System.out.println("메뉴의 이름이 존재하지 않습니다.");
    } else {
        for (int i = index; i < MenuList.size(); i++) {
            MenuList.remove(index);
        }
        System.out.printf("%s메뉴가 삭제되었습니다.", name);
    }
```

기능 시연



기능 시연

MenuIngredient..java

```
public void ingredientMinus() {

    int index1 = IngredientManager.searchIndex("김");

    IngredientManager.ingredientList.get(index1).setNum(IngredientManager.ingredientList.get(index1).getNum() - 1);

    int index2 = IngredientManager.searchIndex("쌀");

    IngredientManager.ingredientList.get(index2).setNum(IngredientManager.ingredientList.get(index2).getNum() - 2);

    int index3 = IngredientManager.searchIndex("단무지");

    IngredientManager.ingredientList.get(index3).setNum(IngredientManager.ingredientList.get(index3).getNum() - 1);

    cnt++;
}
```

Menu.java

```
public void ingredientMinus() {
    super.ingredientMinus();
    info.setNum(info.getNum() - 1);
}
```

메뉴선택시 해당재료 자동감소

SellManager.java

```
// 주문 메서드
// 다른 메뉴도 주문할 때 false 값을 반환.
boolean whatOrderNum(Menu m) {
    boolean addOrder = false;
    System.out.println("몇 개 주문해 ?");
    select = sc.nextInt();
    if (select < 1) {
        System.out.println("1개 이상 주문해");

    }

    else {

        for (int i = 0; i < select; i++) {
            m.ingredientMinus();

        }
    }
```

프로젝트의 개선방향은 무엇인가요?

1. 재료감소의 개수 설정(변경가능하도록)

```
public void ingredientMinus() {
    int index1 = IngredientManager.searchIndex("김");
    IngredientManager.ingredientList.get(index1).setNum(IngredientManager.ingredientList.get(index1).getNum() - 1);
    int index2 = IngredientManager.searchIndex("쌀");
    IngredientManager.ingredientList.get(index2).setNum(IngredientManager.ingredientList.get(index2).getNum() - 2);
    int index3 = IngredientManager.searchIndex("단무지");
    IngredientManager.ingredientList.get(index3).setNum(IngredientManager.ingredientList.get(index3).getNum() - 1);
    cnt++;
}
```

2. 메뉴수정시 재료정보도 수정가능하도록 변경

협업과정에서의 문제점

- **설계도**
  충분한 상의는 하였으나 설계도를 짜는 것이 미숙하여
  틀이 제대로 잡히지 않은 상태에서 시작

- **깃허브**
  깃허브가 익숙하지 않아서
  파일도 한번 날리고.. 또 복구하는 방법은 모르고..

- **코드 통일X**
  Util파일이나 공동으로 쓰이는 변수/메소드의 이름을 통일하지 않아
  마지막에 파일을 합칠때 하나 하나 고쳐야 했음

최종시연

“감사합니다!”